УДК 004.942

Р.І. Дзінько, А.М. Гордійчук, О.І. Лісовиченко

ВИКОРИСТАННЯ МІКРОПОТОКІВ ДЛЯ ЗБОРУ ТА ОБРОБКИ ДАНИХ ПРО СТАН ГВС

Анотація: Розглядається проблема збору і аналізу даних в реальному часі, що є досить специфічною і відзначається такими особливостями, як постійне високе навантаження на систему, необхідність паралельного чи псевдопаралельного виконання програмного коду.

Пропонується підхід до збільшення процесорного часу, що використовується на корисну роботу при використанні мікропотоків, на відміну від послідовного алгоритму, який провокує простої через блокування роботи програми операціями вводу-виводу.

Ключові слова: паралельне виконання програмного коду, збір і аналіз даних, оптимізація використання обчислювальних ресурсів, потоки, мікропотоки.

Вступ

Задача збору і аналізу даних в реальному часі є досить специфічною і відзначається такими особливостями, як постійне високе навантаження на систему, необхідність паралельного чи псевдопаралельного виконання програмного коду. Саме тому оптимізація використання обчислювальних ресурсів для даної задачі є необхідним кроком, на який потрібно звертати увагу.

Ще однією особливістю такого класу задач є наявність великої кількості операцій вводу — виводу, які, як відомо, блокують роботу програми, тобто при виконанні зчитування інформації з датчика певного вузла ГВС буде виконано блокування програми, аж поки дані не будуть отримані. Ситуація ускладнюється тим, що датчики можуть провокувати затримки в надсиланні даних, наприклад, у зв'язку з поломками, чи взагалі можуть вийти з ладу. Таким чином, ми не будемо отримувати реальних даних про ГВС в кожен дискретний період часу, а матимемо "провали".

Виходом з даної ситуації є створення програмного забезпечення з використанням технології багатопотоковості. Потоки, втім, використовують спільні ресурси і накладають блокування на них, таким чином, що один потік не зможе отримати доступ до заблокованого іншим потоком ресурсу. Ще одним недоліком потоків є те, що при створенні нового потоку для цього потоку створюється новий стек пам'яті, що значно обмежує кількість потоків, які можливо створити. Тому такий варіант вирішення не є повним, а лише частковим. Також велика кількість потоків на мало процесорних системах спричиняє "боротьбу" потоків за ресурси, внаслідок чого з'являються значні затримки в виконанні програми.

Одним з варіантів вирішення проблеми є використання мікропотоків.

Мікропотоки — фрагменти програмного коду, що використовуються зазвичай разом з подійно-орієнтованою парадигмою програмування, які можуть виконуватись псевдопаралельно та використовуються з ціллю

підвищення ефективності використання ресурсів мікропроцесорних систем. Навідміну від звичайних потоків (потоків виконання), при створенні мікропотоку не відбувається створення нового стеку. Мікропотоки, також, застосовуються в програмних системах, що характеризуються великими обчислювальними навантаженнями з великою кількістю операцій вводу/виводу даних.

Постановка задачі

Розглянемо ГВС з датчиками збору інформації на її вузлах, що в кожен визначений проміжок часу передають інформацію на модуль збору/аналізу інформації про стан цієї ГВС.

Зображена на рис. 1 ГВС складається з шести оброблюючих ресурсів, до кожного з яких прикріплений датчик, що отримує дані про стан кожного модуля. В визначений період часу (наприклад, кожну мілісекунду) датчик отримує інформацію з ГВМ та передає цю інформацію в модуль збору та аналізу інформації про стан ГВС, використовуючи провідні та безпровідні (резервні та для підвищення надійності) канали зв'язку. Цей модуль отримує інформацію, виконує обробку даних, журналювання (запис даних в базу даних для подальшої статистичної обробки результатів спостережень) та візуалізацію.

Операції вводу виводу спричиняють блокування роботи програми аж до того часу, доки дані з датчиків не буду доступні для зчитування. До цього часу необхідно надіслати запит на отримання даних і очікувати відповіді від датичка. Цей час може використовуватись на виконання корисної роботи. Наприклад, для аналізу та візуалізації даних, чи для отримання даних з інших датчиків.

Традиційний вихід — використання багатопроцесорної системи та багатопотокового програмного забезпечення (ПЗ), але таке рішення є досить дорогим, тому розглянемо реалізацію ПЗ на базі мікропотоків на однопроцесорній системі, а також яким чином мікропотоки дозволяють уникнути простоїв в роботі системи, таким чином оптимізуючи використання обчислювальних ресурсів.

Вмістилищем даних спостереження за математичною моделлю може бути звичайний текстовий чи бінарний файл, а у випадку великих і складних систем — база даних. Як і будь-яке інше вмістилище даних, воно підтримує операції зчитування та запису даних. Якщо ці операції виконуються дуже часто, велика кількість процесорного часу втрачається на те, щоб програма змогла отримати доступ до нього.

Для візуалізації даних в ГВС на модулі збору та аналізу даних краще за все використовувати web-додаток. Web-додатки також являються клієнт-серверними та можуть спричиняти блокування, саме тому при створенні web-додатку слід використовувати мікропотоки. Проблема вибору технології для створення відповідного web-додатку описана в [3].

Ситуація ускладнюється, коли вмістилище даних знаходиться не на тому ж комп'ютері, що і компонент системи, який виконує розрахунки для модельованої системи, а на іншому комп'ютері в локальній мережі.

 ISSN 1560-8956

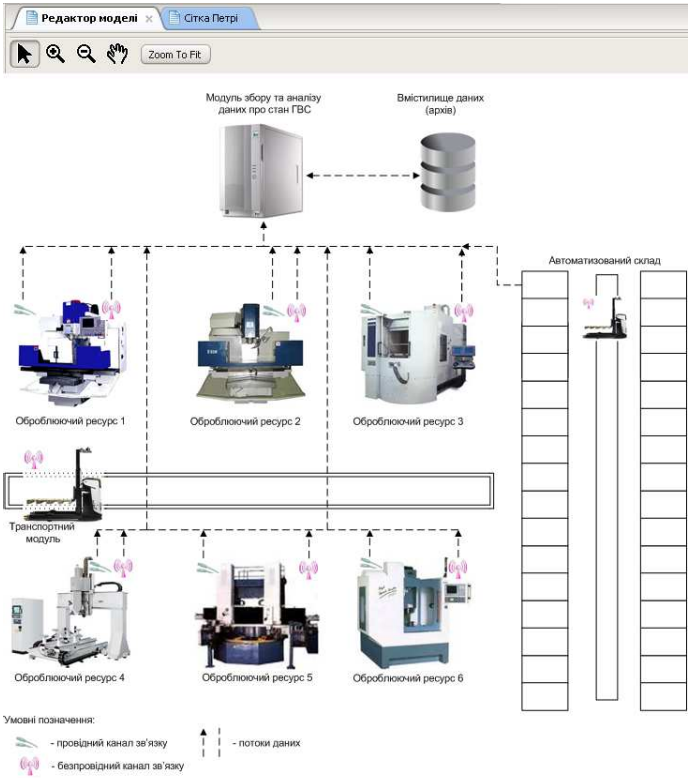

Рис. 1 – Структурна схема досліджуваної ГВС

При послідовному виконанні алгоритму моделюючий компонент повинен чекати, доки стане можливим запис до вмістилища (наприклад, потрібно чекати, доки відкриється з’єднання).

Використання ресурсів процесора при послідовному алгоритмі

Розглянемо діаграму виконання програмного коду для випадку, коли система реалізована як послідовний алгоритм.

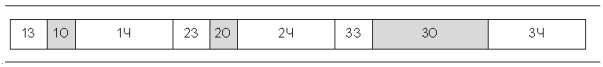

Рис. 2 – Виконання програмного коду при послідовному алгоритмі

На рис. 2 позначений процес виконання програмного коду за деякий період часу у випадку, коли програмна система реалізована як послідовний алгоритм. Кожен блок позначений двома символами. Перший —

номер ГВМ, з яким ведеться робота (див. рис. 1), другий — операція (“З” — запит на отримання інформації з датчика, “О” - очікування отримання інформації, “Ч” - зчитування інформації). Відповідно темними зображені блоки, що означають простої в роботі, світлими — блоки, під час яких виконується корисна робота обчислювальної системи.

Щоб наглядніше оцінити втрати, що зазнає процесор, який застосовується в модулі збору та аналізу інформації про стан ГВС, виконаємо моделювання даної ситуації програмним способом. Дана система являє собою клієнт-серверний додаток, де клієнтами виступають передавачі інформації з оброблюючих ресурсів. Система випадково моделює виведення з ладу датчиків відповідно до їх коефіцієнту надійності (тобто відношення часу, який датчик перебуває в несправності до загального часу роботи ГВС). Приймемо це значення за 95%, тобто 5% часу датчик не в змозі з певних причин передавати інформацію на сервер.

Як бачимо на прикладі очікування даних з третього ГВМ, програма дуже довго простоює, доки датчик не готовий переслати дані в модуль збору та аналізу інформації про стан ГВМ. Якщо датчик вийшов з ладу — цей процес може затягнутись якзавгодно довго, або до того часу, який встановлений як ліміт очікування. В будь-якому випадку програма за цей період часу не виконує корисну роботу.

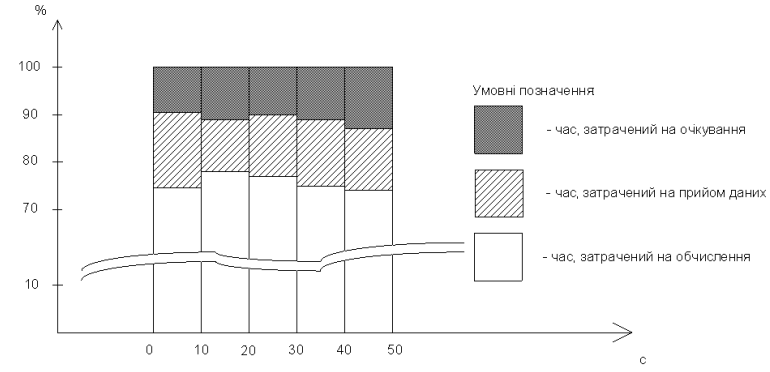

Рис. 3 – Гістограма затрат часу на виконання різних типів задач під час моделювання збору інформації про стан ГВС при послідовному алгоритмі.

З рис. 3 бачимо, що близько 15% робочого часу процесора витрачається на очікування можливості отримання даних з датчика, тобто корисну роботу процесор виконує всього 85% свого робочого часу.

Використання ресурсів процесора при використанні мікропотоків

Розглянемо діаграму виконання програмного коду, коли система реалізована на базі мікропотоків (рис 3). В такому випадку програма являє

 ISSN 1560-8956

собою безкінечний цикл, що оперує з’єднаннями до датчиків. Кожну ітерацію виконується спроба зчитування інформації з поточного датчика; якщо ця можливість відсутня (датчик не готовий пересилати дані), то переходимо до наступного, і т. д..

Рис. 4 – Виконання програмного коду при реалізації алгоритмів на базі мікропотоків

На відміну від представленого на рис. 2 послідовного процесу виконання, мікропотоки не будуть очікувати можливості зчитування даних. Як тільки мікропоток виявиться в ситуації, коли дані з датчика зчитати неможливо, він перейде до спроби зчитати дані з наступного датчика. Час, який використовується на спробу даних є мізерний відносно того, який витрачається на очікування можливості зчитування, тому програма постійно виконує тільки корисну роботу, окрім ситуації, коли всі датчики недоступні. В такому випадку система просто буде опитувати кожен датчик, аж поки той не матиме змоги передати дані.

Таким чином система не має простоїв, тобто в кожен момент часу виконується програмний код, який може виконуватись, і як тільки мікропоток починає простоювати (зчитування даних неможливе), то виконання програми переключається на інший мікропоток.

Мікропотоки дозволяють позбутися простоїв програми, тобто процесорний час використовується практично повністю на виконання корисних задач.

Наприклад, виконаємо моделювання процесу збору та аналізу даних з ГВС як у випадку з послідовним алгоритммом (рис. 3), але з використанням мікропотоків (рис. 5).

Вимірювання показали, що близько близько 0.3% робочого часу процесора витрачається на очікування можливості отримання даних з датчика, тобто корисну роботу процесор виконує 99,7% свого робочого часу, таким чином звільнивши 14.7% часу роботи процесора для виконання розрахунків та зчитування даних.

Розглянемо два підходи до вирішення задачі оптимізації використання обчислювальних ресурсів:

1. підвищення надійності каналів передачі даних та датчиків;

2. заміна послідовного алгоритму виконання програми на алгоритм на базі мікропотоків.

Перший спосіб має сильний недолік – надто великі витрати на високоякісне обладнання, які в більшості випадків не компенсують виграшу, що отримується. Виграш отримуємо тільки на гнучких виробничих ділянках невеликою кількістю оброблюючих ресурсів при великосерійному виробництві.

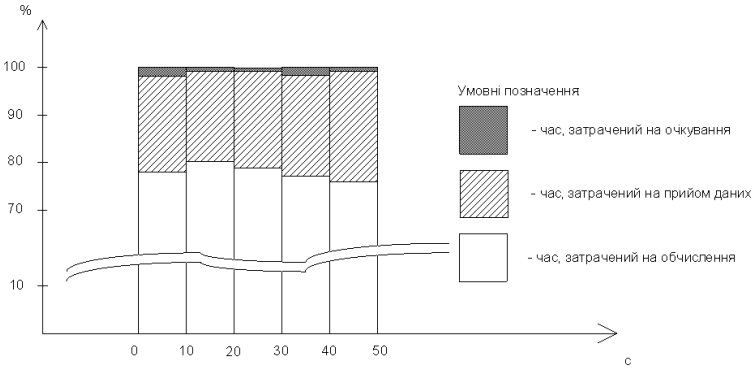

Рис. 5 – Гістограма затрат часу на виконання різних типів задач під час моделювання збору інформації про стан ГВС при використанні мікропотоків.

Розрахунок простоїв процесора

Розглянемо ГВС, що зображена на рис. 1. Дана ГВС складається з n ГВМ, кожен з яким оснащений датчиком збору інформації про стан ГВС. Кожен датчик відправляє блок інформації про стан ГВС з періодом часу (T). Звісно, що жодне обладнання, в тому числі і датчики може піддаватись збоям в процесі роботи чи виходити з ладу. Для простоти обчислень візьмемо коефіцієнт надійності датчика рівним 97% $(K = 0,97)$. Ціллю розрахунків є отримання коефіцієнту корисної дії (ККД) процесора (η), на якому виконується ПЗ для збору/аналізу інформації про стан ГВС. Даний коефіцієнт будемо вираховувати як відношення часу корисної роботи процесора T_K до загального часу роботи ГВС T_3:

$$\eta = \frac{T_K}{T_3} \tag{1}$$

Час корисної роботи можна представити як різниця загального часу роботи ГВС та часу простою процесора T_n :

$$T_K = T_3 - T_n \tag{2}$$

Підставивши формулу (2) в формулу (1) отримаємо:

$$\eta = \frac{T_3 - T_n}{T_3} = 1 - \frac{T_n}{T_3} \tag{3}$$

Для отримання інформації з кожного датчика модуль аналізу інформації про стан ГВС повинен мати термін очікування (у випадку послідовного алгоритму), по закінченні якого він повинен перейти до зчитування інформації з іншого датчика. Цей термін обирається довільно і є константою T_o. Спроба зчитати інформацію без очікування є величиною дуже малою порівняно з іншими, що використовуються в даній задачі, тому

нею можна знехтувати. Час зчитування даних залежатиме від розміру блоку даних, що передається датчиком під час одного зчитування.

Окрім зчитування інформації модуль збору та аналізу інформації виконує також перетворення і підготовку інформації для візуалізації та збереження до вмістилища даних для проведення подальшої статистичної обробки. Використання процесорного часу на ці дії є корисним часом.

Оскільки, ми не можемо передбачити, чи почне зчитувати модуль аналізу інформацію з датчика саме в період його відмови, введемо коефіцієнт ймовірності зчитування даних з датчика в період його відмови (R).

В разі використання алгоритму з використанням мікропотоків, оскільки при відмові датчика на передачу даних, час не буде витрачатись на простої, а витратиться на зчитування інформації з наступного дачтика, коефіцієнт корисної дії буде таким, що максимально наближається до одиниці, тобто $\eta \approx 1$.

Розрахуємо час простою при послідованому алгоритмі, додавши до формули розрахунку коефіцієнт R таким чином, щоб змоделювати ситуацію, коли чим більша ймовірність попасти в період відмови, тим більший час простою.

$$T_n = n \cdot (1 - K) \cdot T_3 \cdot R \tag{4}$$

Підставивши формулу (4) в формулу (3) отримаємо наступний результат:

$$\eta = 1 - \frac{n \cdot (1 - K) \cdot T_3 \cdot R}{T} = 1 - n \cdot (1 - K) \cdot R \tag{5}$$

Використаємо для тестових розрахунків ГВС представлену на рис. 1. Дана ГВС отримує інформацію з восьми датчиків (6 на оброблюючих вузлах та 2 на автоматизованих транспортних модулях). Приймемо коефіцієнт надійності датчиків 97% і змоделюємо роботу системи протягом 1 год (3600 с). Виконаємо розрахунки при 2-х значеннях коефіцієнта R, які є границями діапазону допустимих значень для цієї величини $(0..1 - K)$. В такому випадку:

$$n = 8; \quad K = 0.97; \quad T_3 = 3600; \quad R_1 = 0; \quad R_2 = 0.03$$

Розраховуємо коефіцієнт корисної η:

$$\eta_1 = 1 - 8 \cdot (1 - 0.97) \cdot 0 = 1 \tag{6}$$

$$\eta_2 = 1 - 8 \cdot (1 - 0.97) \cdot 0.03 = 0.9928 \tag{7}$$

Проаналізувавши отримані результати, бачимо, що у випадку використання послідовного алгоритму за годину роботи ГВС 26 секунд процесорного часу буде втрачено на простій. У випадку з даною ГВС це не така вже й велика величина, проте набагато більші втрати можна здобути на дещо більшій ГВС. Так, наприклад, якщо ГВС для роботи використовує 100 датчиків замість 8-ми, як було подано в розрахунках, ККД

становитиме 91%, тобто процесор потратить більше 5-ти хвилин часу на простої.

Висновки

Аналізуючи діаграми виконання програмного коду при двох описаних підходах (рис. 2 і рис. 3) та розрахунки коефіцієнту корисної дії процесора при різних алгоритмах, ми бачимо очевидне збільшення кількості процесорного часу, що використовується на корисну роботу при використанні мікропотоків, на відміну від послідовного алгоритму, який провокує простої через блокування роботи програми операціями вводу-виводу.

Оскільки, точно підрахувати час простою для ГВС неможливо, адже цю величину через випадкову природу періодів виходу датчиків з ладу, можливо оцінити тільки статистично, бачимо, що для кожної ГВС ККД роботи модуля збору/аналізу даних про стан ГВС при послідовному алгоритмі буде вар’юватись в певному діапазоні. При достатньо великій кількості датчиків процесор буде простоювати повністю.

Також поряд з підвищенням ефективності використання обчислювальних ресурсів в процесів роботи була виявлена ще одна перевага мікропотоків — надійність та стабільність програмного забезпечення. Оскільки модуль збору та аналізу інформації повинен виконувати задачі не тільки зчитування та запису даних до вмістилища інформації, а також аналіз даних та візуалізацію даних. Ці додаткові задачі теж будуть перериватись при послідовному виконанні. У випадку з мікропотоками — процесорний час буде повністю виділятись і на всі задачі.

Література

1. MEANS: A Micro-thrEad Architecture for Network Server (2008) - Yingchun Lei, Wen Zhang, Yili Gong, Huyin Zhang http://www.computer.org/portal/ web/ csdl/ doi/ 10.1109/ PDP.2008.17 -
2. Reconfigurable computing based on universal configurable blocks-a new approach for supporting performance- and realtime-dominated applications (2000) - Siemers, C.; Siemers, S. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp ?arnumber =824328
3. Вибір технології для створення web-систем різного рівня складності / Дзінько Р.І., Лісовиченко О.І., Гордійчук А.М. // Адаптивні системи автоматичного управління. – 2009. - 15(35). – С. 22-30.

Отримано 07.03.2011 р.

 ISSN 1560-8956